Das Menschliche Gehirn – nur ein Rechner?

Wolfgang Maass

Institut für Grundlagen der Informationsverarbeitung
Technische Universität Graz
Klosterwiesgasse 32/2, A-8010 Graz, Austria
email: maass@igi.tu-graz.ac.at
http://www.cis.tu-graz.ac.at/igi/maass/

1 Einführung

Die Vorstellung, daß das menschliche Gehirn im Prinzip durch einen künstlichen Rechner ersetzt werden könnte, wirkt nicht sehr plausibel. Wir alle haben ja eine recht gute Vorstellung davon, was ein künstlicher Rechner (= Computer) kann, oder vielmehr nicht kann:

– man muß ihm vorher alles haargenau erklären (d.h.: einprogrammieren) bevor er eine Aufgabe ausführen kann

– er wird nicht aus Erfahrung klug: wenn wir sein Programm nicht ändern, wird er denselben Fehler wieder und wieder machen

– er kann sich nicht selbständig weiterentwickeln, und ist in keiner Weise kreativ

– er hat keine Gefühle oder Bewußtsein

– er ist instabil und anfällig: wir alle wissen wie gerne ein Rechner "abstürzt", und wie anfällig er ist gegenüber eingeschleusten Computerviren sobald er mit anderen Rechnern in Kontakt tritt.

Obwohl die kommerziell vertriebenen Rechner im Laufe der Zeit immer schneller und kleiner wurden, haben sie sich erstaunlich wenig verbessert in bezug auf die angeführte Mängelliste. Ich werde in diesem Aufsatz argumentieren, daß dies gar nicht so erstaunlich ist, weil diese Mängel auf grundlegenden in der Theorie der Rechner verankerten "Planungsfehlern" beruhen, die wiederum tief in althergebrachten aber zum Teil fragwürdigen Vorstellungen über Maschinen und menschliche Intelligenz verwurzelt sind.
Ich werde diese Kritik anhand des wohlbekannten Turing-Tests präzisieren, den der geniale Mathematiker und Computer-Pionier Alan Turing vor fast genau 50 Jahren veröffentlicht hatte [Turing, 1950]. Er schlug vor, folgendes Kriterium dafür zu wählen, ob ein künstlicher Rechner "intelligent" sei:

> *Ein Rechner sollte dann als intelligent bezeichnet werden, wenn wir als Mensch bei einem beliebigen Frage-und-Antwort Spiel, das über eine elektronische Verbindung (zum Beispiel per email) durchgeführt wird, nicht unterscheiden können, ob am anderen Ende der Leitung dieser Rechner oder ein anderer Mensch sitzt.*

Dieser Test ist gedanklich sehr attraktiv, und er hat unsere Vorstellung von "maschineller Intelligenz" nachhaltig geprägt. Erst wenn man etwas länger über diesen Test nachdenkt, fällt einem auf, daß er stark beeinflußt ist von einem Intelligenzbegriff, dessen Akzeptanz heute im Schwinden begriffen ist. Der Turing-Test beruht auf der Vorstellung von Intelligenz als der Fähigkeit, bei einer Unterhaltung mit Menschen, die sich selbst für intelligent halten, mitreden zu können, ohne dabei unangenehm aufzufallen. Ein offensichtlicher Defekt dieses Intelligenzbegriffs ist, daß er jemanden, der quasi als Schauspieler typische Sprachspiele von selbsternannten intelligenten Menschen mitspielen kann, ohne notwendigerweise zu *verstehen* worüber er redet, als intelligent bezeichnen würde. Weitere Defekte des im Turing-Test formalisierten Intelligenzbegriffs sind dessen Festlegung auf verbale Kommunikation, sowie die Vernachlässigung von anderen wichtigen Aspekten der Intelligenz, wie zum Beispiel der "praktischen Intelligenz", oder allgemeiner der Fähigkeit "sich zu helfen zu wissen", also der Fähigkeit unvorhergesehene Probleme in kreativer Weise zu lösen.
Bedauerlicherweise hat die dem Turing-Test zugrunde liegende einseitige Vorstellung von Intelligenz nicht nur viele Generationen von Schülern und Studenten, sondern auch unsere traditionellen Rechner geprägt. Der Zusammenhang zwischen der im Turing-Test formalisierten Vorstellung von Intelligenz und den Unzulänglichkeiten unserer gegenwärtigen Rechner liegt auf der Hand: man kann Rechner jahrzehntelang weiterentwickeln und deren Abschneiden im Turing-Test verbessern, ohne daß die so enstehenden Rechner notwendigerweise große Fortschritte hinsichtlich der am Anfang angeführten Mängelliste aufweisen würden. In diesem Sinn habe ich am Beginn dieses Artikels etwas überspitzt argumentiert, daß man die angeführten Mängel unserer gegenwärtigen Rechner als Planungsfehler auffassen kann.[1]
Als Gegenpol möchte ich hier einen alternativen Test für die "Intelligenz" eines Rechners vorschlagen:

> *Die Intelligenz eines Rechners wird daran gemessen wie lange er unter wechselnden und teilweise unvorhersehbaren und widrigen Umständen "überlebt".*

Dabei kann man "überleben" eines Rechners in geeigneter Kontext-bezogener Weise definieren (zum Beispiel als Aufrechterhalten seiner üblichen Rechnerleistung ohne Absturz.) Je länger die Überlebensdauer eines Rechners ist und umso widriger und stärker schwankend die Umstände waren, denen der Rechner getrotzt hat,

[1] Eine genauere Betrachtung ergibt ein differenziertes Bild. Vom Gesichtspunkt der Computerindustrie ist es logisch und sinnvoll die Rechnerentwicklung so anzulegen, daß der Kunde alle paar Jahre seine software und hardware ersetzen muß. Unter diesem Gesichtspunkt wäre es kontraproduktiv, den Schwerpunkt in der Rechnerentwicklung umzustellen auf die Entwicklung von Rechnern die sich selbständig auf Änderungen in ihrer Umgebung einstellen und ihre eigene Leistungsfähigkeit weiterentwickeln, und dabei möglicherweise sogar eine "Individualität" entwickeln, an der der Kunde hängt.

umso "intelligenter" würde ich einen solchen Rechner nennen. Dabei kann man bei den "unvorhersehbaren und teilweise widrigen Umständen", denen der Rechner ausgesetzt wird, an das Einspielen von fehlerbehafteten neuen Programmen, Unmutsäußerungen des menschlichen Benutzers oder anderer Maschinen, menschliche Bedienungsfehler, sich einschleichende Computerviren, oder sogar an Stromausfall, Wasserrohrbruch, oder Erdbeben denken.
Gemäß diesem Kriterium sind selbst primitive Lebewesen wie Käfer oder Stubenfliegen in gewisser Weise intelligenter als unsere gegenwärtigen Rechner, weil sie vielen physischen Bedrohungen geschickt ausweichen können. Diese Folgerung ist kein Widerspruch, sondern sie macht die Schmalspurigkeit des im Turing-Test formalisierten Intelligenzbegriffs besonders deutlich: Intelligenz bedeutet ja auch Gefahren rechtzeitig zu erkennen, sich mit beschränkten Mitteln in der Not zu helfen wissen.
Offensichtlich ist das "timing", oder genauer gesagt: die Rechengeschwindigkeit, ein wesentlicher Bestandteil des alternativen Intelligenzbegriffs, denn es nützt nichts, eine perfekte Lösung in einer Minute Rechenzeit zu finden, wenn eine das Überleben ermöglichende Reaktion in "Echtzeit", also zum Beispiel innerhalb 1/10 Sekunde erforderlich ist. Das bedeutet, daß die Gültigkeit der Turing Maschine[2] als geeignetes mathematisches Modell für intelligente Rechner in Frage gestellt wird. Die Turing Maschine ist ein Modell für einen *sequentiellen* Rechner, in dem in jedem Zeittakt jeweils nur eine Rechenoperation ausgeführt werden kann. Daher können nur einige extrem einfache Rechenaufgaben auf einer Turing Maschine in "Echtzeit", also sagen wir in höchstens 10 Zeittakten, ausgeführt werden.[3] Dagegen sind biologische Nervensysteme dadurch ausgezeichnet, daß sie einige spezielle Aufgaben der Informationsverarbeitung, deren schnelle Lösung "überlebenswichtig" ist, in Echtzeit ausführen können. Dies erreichen sie dadurch, daß sie solche Aufgaben an eine große Anzahl von "Prozessoren" verteilen, die "parallel" an der Problemlösung arbeiten. Dabei hilft ihnen die Architektur ihrer Schaltkreise, deren Grundbausteine – die Neuronen – Inputs von 5000 – 10000 anderen Neuronen bekommen, und ihren Output an eine etwa gleichgroße Zahl anderer Neuronen parallel (d.h. gleichzeitig) übermitteln können. Bei der Konstruktion einer gemäß dem alternativen Intelligenz-Kriterium "intelligenten" Maschine muß man sich entscheiden, welche komplexen aber "überlebenswichtigen" Berechnungen auf ihr unbedingt in Echtzeit durchführbar sein müssen. Da auf einer Turing Maschine *keinerlei* komplexe Berechnungen in Echtzeit durchgeführt werden können, ist sie

[2]Die Turing Maschine ist ein von Turing vorgeschlagenes mathematisches Rechnermodell, das "universell" ist in dem Sinn, daß es jedes andere Modell für einen digitalen und deterministischen Rechner simulieren kann (wobei die Turing Maschine aber wesentlich mehr Rechenschritte benutzen darf als der simulierte Rechner), siehe [Sipser, 1997, DePauli und Weibel, 1997]

[3]P ist ein mathematisches Modell für die Klasse aller tatsächlich auf einem Computer lösbaren Berechnungsprobleme. Ein Problem ist dann in der Klasse P wenn es einen Algorithmus zur Lösung dieses Problems gibt, und zusätzlich die benötigte Rechnerzeit nur polynomiell (also nicht "explosionsartig" wie bei der Exponentialfunktion $n \mapsto 2^n$) mit der Input-Länge n anwächst. NP ist die Klasse aller auf einer fiktiven "nichtdeterministischen" Turing Maschine in polynomieller Rechenzeit lösbaren Probleme. Von vielen für die Praxis besonders wichtigen Berechnungsprobleme, zum Beispiel aus den Bereichen Operations Research, Scheduling, Kryptographie, ist bekannt, daß sie zu der Klasse NP gehören. Weil nur $P \subseteq NP$ bekannt ist, besagt das aber nicht, daß diese Probleme tatsächlich auf einem Computer lösbar sind, also zur Klasse P gehören.

als mathematisches Modell für derart "intelligente" Maschinen ungeeignet.
Ich werde im Abschnitt 2 dieses Artikels einige Forschungsansätze vorstellen, die zeigen, daß es durchaus schon Ideen und Bausteine gibt für einen neuartigen Typ von Rechnern, die gemäß dem alternativen Intelligenz-Kriterium als "intelligent" bezeichnet werden könnten. Dies läßt uns erahnen wieviel mehr ein Rechner im Prinzip kann – verglichen mit dem traditionellen Begriff eines Rechners. Gleichzeitig werden wir Indizien für eine möglicherweise überraschende Entwicklung sehen: Während man früher das menschliche Gehirn als eine Variante von Idealisierungen vorhandener Rechner (also zum Beispiel von Turing Maschinen) zu verstehen suchte, hat sich mittlerweile der Spieß umgedreht: viele bahnbrechende neue Ideen in der Informatik haben ihre Wurzel in Erkenntnissen und Vermutungen über die Arbeitsweise biologischer Nervensysteme.
Wenn ich im 3. Abschnitt dieses Artikels zurückkomme auf die Frage "Das menschliche Gehirn – nur ein Rechner?" werden wir sehen, daß das eigentlich Problematische an dieser Vermutung das Wörtchen "nur" ist, weil ein Rechner, der alles wirklich "Rechnermögliche" ausschöpft, ein ungleich anderes Wesen ist als die uns bisher geläufigen Rechner. Gleichzeitig werden wir sehen, daß eine positive Antwort auf die diesem Aufsatz zugrundeliegende Frage nicht aufgefaßt werden könnte als ein Sieg der Natur- und Ingenieurwissenschaften über die Geisteswissenschaften, der Maschine über die belebte Materie. Vielmehr wird sichtbar, daß die Entwicklung wirklich "intelligenter" Maschinen dazu führt, daß die Grenzen zwischen beiden Bereichen verwischt werden.

2 Vorboten eines neuen Typs intelligenter Rechner

Die Entwicklung von Rechnern, die gemäß dem im ersten Abschnitt vorgestellten alternativen Intelligenz-Kriterium als intelligent zu bezeichnen wären, erfordert ein radikales Umdenken. Ich werde in diesem Abschnitt einige erfolgversprechende Ansätze skizzieren.

2.1 Vom Logiker zum Käfer: Wechselnde Vorbilder für Maschinelle Intelligenz

Lange Zeit war bei der Entwicklung maschineller Intelligenz das Hauptziel, einen maschinellen Theoretiker zu schaffen, also eine Maschine, die ein theoretisches Modell ihrer Umwelt aufbaut, und die zur Lösung eines konkreten Problems Strategien sucht für die sie logisch herleiten kann, daß sie optimal sind. Bei diesem Ansatz ist man auf zwei Probleme gestoßen:

- Solange der Rechner keine begleitende Intuition für sein theoretisches Welt-Modell besitzt, muß er bei der Berechnung von sachgerechten Problemlösungen mehr oder weniger *blind suchen* bis er *zufällig* auf eine Strategie stößt, die "paßt". Für ihn ist ja jede mögliche Strategie oder "Aktion" nur eine Zeichenreihe, die genauso gut in Chinesisch kodiert sein könnte, weil er ihren Inhalt sowieso nicht versteht. Da der Suchraum für mögliche Problemlösungen in der Regel exponentiell groß ist in der Komplexität n des Problems (wobei man

n als die Anzahl der "Freiheitsgrade" oder Teilschritte von möglichen Strategien auffassen kann), führt dies schon bei relativ einfachen Problemen (also zum Beispiel für $n = 100$) zu einer erforderlichen Anzahl von Rechenschritten der Größenordnung 2^{100}. Die allerschnellsten gegenwärtig existierenden Rechner können weniger als 2^{44} Rechenschritte pro Sekunde ausführen. Zur Ausführung von 2^{100} Rechenschritten würde solch ein Rechner 2^{56} Sekunden, also ca. 1 000 000 000 Jahre benötigen.

Ein wesentlicher Durchbruch könnte hier im Prinzip erzielt werden, wenn man beweisen könnten, daß die Komplexitätsklassen P und NP zusammenfallen.[4]An diesem Problem wird in der Theoretischen Informatik und Mathematik seit 30 Jahren intensiv gearbeitet, ohne daß eine Lösung am Horizont sichtbar wäre. Die gängige Vermutung besagt allerdings, daß die Komplexitätsklassen P und NP *nicht* zusammenfallen, was bedeuten würde, daß es bei wichtigen Problemen des maschinellen logischen Denkens keine wesentlich schnellere Alternative zum blinden Suchen nach passenden Schlußketten gäbe.

- Ein weiteres und möglicherweise noch schwierigeres Problem ist das folgende: Wir als Menschen schaffen es in der Regel halbwegs richtige Entscheidungen im Alltagsleben zu treffen, obwohl sowohl das uns zur Verfügung stehende Hintergrundwissen[5] als auch die unmittelbar für die Entscheidung relevanten Sinneseindrücke in der Regel vieldeutig und oft sogar widersprüchlich sind. Diese Komplikation wird uns meist gar nicht bewußt solange wir nicht darüber nachdenken. Sie führt aber zur bedauerlichen Tatsache, daß die Kalküle der mathematischen Logik und die bisher vorhandenen Methoden des maschinellen Beweisens nicht geeignet sind um einer Maschine in einer natürlichen Umgebung sachgerechte Aktionen vorzuschlagen.[6]

Ein besonders interessantes Anwendungsgebiet für maschinelle Intelligenz ist die Robotik. Die beiden genannten Probleme haben bewirkt, daß ein mit traditioneller künstlicher Intelligenz, also mit Hintergrundwissen und logischen Schlußweisen,

[4]Ein weiterer Grund weshalb Turing Maschinen keinen geeigneten Vergleichsmaßstab für das menschliche Gehirn liefern ist ein rein mathematischer: Turing Maschinen sind als mathematisches Rechnermodell nur dann von Interesse, wenn beliebig lange Bit-Folgen als Inputs zu verarbeiten sind, weil sie sich sonst wie endliche Automaten verhalten. Es haben aber alle durch ein menschliches Gehirn verarbeiteten Inputs eine durch eine fixe Zahl beschränkte Anzahl von Bits.

[5]Wir bezeichnen hier mit Hintergrundwissen das bei genauerem Hinsehen sehr umfangreiche Wissen, das uns – ohne daß wir uns dessen in der Regel bewußt werden – hilft, zielführende Entscheidungen zu treffen. Um ein ganz einfaches Beispiel zu nennen: wenn wir die Route planen, um einen Raum zu durchqueren, benutzen wir unbewußt vielfältiges Hintergrundwissen, zum Beispiel, daß Möbel nicht wirklich zweidimensional sind, wie sie auf unserer Retina erscheinen, daß sich Beleuchtung, Fenster- und Türstellungen schnell ändern können, aber Möbel keine spontanen Sprünge ausführen, und daß wir etwaige Stufen am Boden aber keine Kaffeetische betreten dürfen. Das klingt alles recht einfach, aber erklären Sie dieses Hintergrundwissen einmal Ihrem PC, ohne sich in Widersprüche oder Mehrdeutigkeiten zu verwickeln!

[6]Die hier angesprochene Schwierigkeit bewirkt interessanterweise, daß das berühmte Gödel'sche Theorem über die Unbeweisbeweisbarkeit der Widerspruchsfreiheit eines formalen Systems S mit Mitteln dieses formalen Systems S *kein* zusätzliches Hindernis für praktische maschinelle Intelligenz bedeutet. Die in diesem Kontext relevanten formalen Systeme S sind sowieso nicht widerspruchsfrei.

ausgestatteter autonomer mobiler Roboter sich in einer nicht speziell für ihn präparierten Umgebung in der Regel nur peinlich langsam und nicht besonders zufriedenstellend bewegt.
Ein überraschender Neuansatz zur Lösung dieses Problems, der heute oft als "Neue Robotik" bezeichnet wird, wurde der staunenden Fachwelt im Jahr 1986 von Rodney Brooks[7]vom MIT in Cambridge (USA) vorgestellt. Genaugenommen waren einige dieser revolutionären Ideen schon vorher von einem krassen "Außenseiter" in der Robotik, dem Neurophysiologen Valentino Braitenberg, vorgeschlagen worden.[8] Als Alternative zu den damals besten Robotern, die große Rechner mit sich herumschleppten und lange darüber "nachdenken" mußten wie sie einem Hindernis ausweichen sollten, stellte Rodney Brooks kleine käferartige Leicht-Roboter vor, die flink durch die Gegend wieselten und Hindernisse geschickt umgingen. Ihre Rechner-Architektur beruht auf einem neuen Prinzip, das technisch als Subsumptionsarchitektur[9] bezeichnet wird. Hier versucht man nicht mehr, ein immer komplizierter werdendes Modell der Wirklichkeit im Rechner nachzubauen und bei Entscheidungen zu befragen. Stattdessen besteht das "Innenleben" dieser neuen Wesen aus einem geschickt koordinierten Bündel von spezialisierten "Reflexen", zum Beispiel dem Reflex ein unmittelbar vor ihm befindliches Hindernis zu umgehen, oder dem Reflex eine Ladestation aufzusuchen, sobald die Spannung am Akku des Roboters eine Schwelle unterschreitet, oder dem Reflex ein bestimmtes vom Benutzer vorgegebenes Ziel aufzusuchen. Das Prinzip von Brooks' Subsumptionsarchitektur besteht darin, daß zunächst jeder einzelne dieser Reflexe für sich möglichst direkt durch geeignete Verbindungen von den "Sinnesorganen" (d.h. von Lichtsensoren, sonaren Sensoren, Kameras oder Laser-Abstandsmessern) zu den Motoren des Roboters in möglichst einfacher und stabiler Weise implementiert wird (mehr dazu im nächsten Unterabschnitt). Für den Fall, daß verschiedene dieser Reflexe zu gegensätzlichen Motor-Kommandos führen, werden solche Konflikte intern in einer Weise gelöst, die das langfristige "Überleben" des Roboters optimiert. Zum Beispiel: wenn die Akku-Spannung niedrig ist, aber ein Hindernis auf dem direkten Weg zur Ladestation auftaucht, dann ist es momentan wichtiger diesem Hindernis auszuweichen als zu versuchen, mit "dem Kopf durch die Wand" auf dem direktesten Weg zur Ladestation zu gelangen.
Der geschilderte Neuansatz von Rodney Brooks ist aus der gegenwärtigen Robotik nicht mehr wegzudenken. In Schwierigkeiten gerät dieser Ansatz dort, wo die Vielfalt der zu kontrollierenden Reflexe oder die Kompliziertheit der zu bewältigenden Aufgaben es dem Ingenieur nicht mehr erlauben, die Regeln für ein geeignetes Zusammenspiel der Reflexe mittels seiner eigenen Intuition festzulegen. Ein menschlicher Betrachter dieser Problematik ist vielleicht erinnert an nicht ganz unähnliche Konflikte zwischen kurz- und langfristigen Zielen verschiedener Art, die sich – teilweise unbewußt – im Menschen abspielen.[10]

---

[7]http://www.ai.mit.edu/people/brooks/

[8]Sein Buch "Vehicles: Experiments in Synthetic Psychology" [Braitenberg, 1984] ist noch heute überaus lesenswert und anregend. Eine Zusammenfassung der wichtigsten Abschnitte ist online erhältlich von http://www.cis.tu-graz.ac.at/igi/STIB/WS98/gruppe3/welcome.html.

[9]Man denke zum Beispiel an die Konflikte zwischen Es, Ich und Überich in der Psychoanalyse von Sigmund Freud.

[10]http://www.ai.eecs.umich.edu/cogarch0/subsump/index.html

Ein gemäß der Subsumptionsarchitektur konstruierter Roboter ist zwar einem relativ “primitiven” Lebewesen wie einem Käfer sehr viel ähnlicher als einem komplexen Lebewesen, aber im Unterschied zu einem gemäß der traditionellen künstlichen Intelligenz konstruierten Roboter haben einige dieser Reflexe eine unmittelbar erfahrbare *Bedeutung* für ihn: die gegenwärtige Akkuspannung ist zum Beispiel nicht eine Zahl wie jede andere, sondern sie ist ein unmittelbarer Indikator dafür wie lange er noch umherfahren kann bevor er eine Ladestation aufsuchen muß. Ebenso ist ein hoher Meßwert von Infrarot-Sensoren in Fahrtrichtung ein recht sicheres Vorzeichen für eine unmittelbar bevorstehende Kollision. In diesem Sinn haben diese Sensor-Werte eine unmittelbare *Bedeutung* für den Roboter. Sobald aber intern kodierte Informationen eine direkte Bedeutung für die Existenz einer Maschine haben, erfordert es nur einen kleinen zusätzlichen Schritt, daß die Maschine Gefühle zeigt, also zum Beispiel Freude über eine Akku-Ladung oder Sorge, falls eines seiner Räder nicht mehr die gesendeten Dreh-Befehle ausführt (siehe zum Beispiel die ausführliche Diskussion in Kapitel 2 von [Picard, 1997]).

Zugegebenermaßen sind die angesprochenen praktischen Probleme, wie die Vermeidung von Kollisionen, recht primitiv, verglichen etwa mit den abstrakten Überlegungen eines Logikers oder Mathematikers. Vielleicht haben beide Bereiche aber trotzdem etwas miteinander zu tun: Hat nicht fast jeder Logiker und Mathematiker — bewußt oder unbewußt — räumliche Bilder vor Augen, selbst bei vollkommen abstrakten Gedankengängen, also auch beim *formalen Denken*? Ist dieser Effekt vielleicht ein Überbleibsel des “Käfers in uns”, oder genauer gesagt: der über Millionen von Jahren der Evolution angesammelten Erfahrungen im Umgang mit 3-dimensionalen Objekten? Und liegen die zahlreichen Mißerfolge beim Versuch kreatives formales Denken (zum Beispiel: maschinelles Beweisen) von Rechnern durchführen zu lassen, vielleicht daran, daß man zu geradlinig versucht hat, den Rechner wirklich rein “formal”, also ohne begleitende quasi-räumliche Anschauung, denken zu lassen — was selbst uns Menschen kaum gelingt?

Schließlich möchte ich anmerken, daß der Paradigmenwechsel “vom Logiker zum Käfer” in der Robotik weniger perfekt ist als ich es bisher dargestellt habe. Probleme treten schon dann auf, wenn man in einem komplexen Roboter das Zusammenspiel seiner “Reflexe” so gestalten möchte, daß sein “Überleben” optimiert wird. Man ist versucht, dem Roboter hierfür Hintergrundwissen über die relative Wichtigkeit einzelner Reflexe einzuprogrammieren, aus denen er sich sinnvolle Koordinationsregeln *logisch* herleiten kann (zum Beispiel: “Auch wenn die Akku-Spannung niedrig ist, müssen Hindernisse auf dem Weg zur Ladestation umgangen werden.”).

Bei Käfern und anderen Lebewesen ist das Zusammenspiel der Reflexe im Laufe der *Evolution* über Millionen von Jahren hinweg soweit optimiert worden, daß sie damit überleben können. Diese Methode hat man auch in der gegenwärtigen Robotik erfolgreich eingesetzt: Indem man eine “Evolution” künstlich simuliert, also Teilstücke der Reflex-Koordination der überlebensfähigsten Exemplare der gegenwärtigen Generation von Robotern in verschiedenen Weisen rekombiniert (“Kreuzung”) und lokal zufällig verändert (“Mutation”). Aus der so entstehenden neuen Generation von Robotern wählt man wiederum die überlebensfähigsten aus und iteriert dann das geschilderte Verfahren. Schöne Beispiele von Anwendungen dieser sogenannten “genetischen Algorithmen” in der Robotik werden zum Beispiel

in den Arbeiten der Robotik-Gruppe der EPFL Lausanne geschildert.[11]
Der wesentliche Nachteil dieser Technik besteht darin, daß sie in der Praxis erfordert, die individuell Überlebensfähigkeit von Hunderten und Tausenden verschiedener Roboter-Varianten zu evaluieren. Das nimmt sehr viel Zeit in Anspruch. Daher ist es sinnvoll, diese Evolutionstechnik zu ergänzen durch schnellere Lernmethoden, die schrittweise die Überlebensfähigkeit eines *einzelnen* Roboter-Individuums verbessern können. Solche Techniken werden im nächsten Unterabschnitt skizziert.
Am Rande möchte ich anmerken, daß beide Techniken im Prinzip natürlich auch im Bereich nicht-mobiler Rechner anwendbar sind, also dort wo die "Bedrohungen" – abgesehen von einem Stromausfall – nicht physischer Natur sind, und wo "überleben" für einen Rechner bedeutet, möglichst lange eine hohe Rechenleistung ohne "Absturz" aufrecht zu erhalten.

2.2 Maschinen, die aus ihren eigenen Erfahrungen lernen

Ein wesentlicher Bestandteil des in unserer Gegenwartskultur verankerten Begriffs der Maschine ist die Vorstellung, daß das Verhalten einer Maschine insofern "trivial" ist, als sie nur das ausführen kann, was ihr von ihrem menschlichen Erbauer vorher einprogrammiert wurde. Diese Vorstellung ist veraltet, weil es in den Forschungslabors[12], in der Industrie, und sogar schon in der Unterhaltungselektronik[13] eine Reihe von Programmen gibt, die es Rechnern ermöglichen, aus ihrer eigenen Erfahrung zu lernen. Insbesondere können solche lerndenden Maschinen eine durch ihre vorhergehenden Erfahrungen geformte *Individualität* hervorbringen, die selbst von ihrem menschlichen Erbauer nicht vollkommen vorhersehbar ist. Das einzige, was der menschliche Erbauer einer solchen Maschine kennt, ist deren *Lernalgorithmus*, also das von ihm einprogrammierte Verfahren, mit dem die Maschine neue Verhaltensmuster aufgrund von vorhergehenden Erfahrungen entwickeln kann. Eine Vielfalt solcher Lernalgorithmen ist inzwischen bekannt [Weiss and Kulikowski, 1991, Mitchell, 1997]. Viele der in den Entwicklungslabors bisher erfolgreichsten Lernalgorithmen sind den Lernmechanismen von lebendigen Organismen abgeschaut worden.[14] Als Beispiele erwähne ich das *Reinforcement Lernen*[15] [Sutton and Barto, 1998], sowie das Lernen mittels *künstlicher Neuronaler Netzwerke* [Arbib, 1995]. Das Reinforcement Lernen ist von besonderem Interesse im Kontext des im ersten Abschnitt geschilderten alternativen Intelligenzkriteriums, weil es direkt auf dem Prinzip aufbaut, gegenwärtige Aktionen jeweil so zu wählen, daß die langfristige Belohnung (zum Beispiel: langes Überleben) ma-

[11] http://lamiwww.epfl.ch/lami/team/mondada/index.html; http://diwww.epfl.ch/lami/team/floreano/

[12] http://www.cs.bham.ac.uk/~anp/ai_ml.html; http://www.sgi.com/Technology/mlc/; http://www.ics.uci.edu/AI/ML/MLDBRepository.html; http://www.cis.tu-graz.ac.at/igi/STIB/WS98/

[13] http://www.creatures2.com

[14] Anzumerken ist, daß gesicherte und allgemein anerkannte Kenntnisse über neurobiologische Mechanismen des Lernens im menschlichen Gehirn kaum vorhanden sind, und daß es sich bei den bisher in künstlichen Rechnern realisierten Lernmethoden um Lernmechanismen relativ einfacher Tiere handelt (siehe [Arbib, 1995]).

[15] genauer gesagt: über den Wert ihres gegenwärtigen Zustands hinsichtlich der von ihr angestrebten Ziele

ximiert wird. Ein weiterer interessanter Aspekt des Reinforcement Lernen ist die Tatsache, daß es dem Rechner, bzw. dem davon gesteuerten Roboter, beibringt, ständig abzuschätzen, wie weit er von einem vorgegebenen Arbeitsziel noch entfernt ist. Wenn man möchte, kann man das auf diese Weise erzielte Wissen der Maschine über ihre eigene Situation [16] als rudimentäre Vorform eines maschinellen Bewußtseins bezeichnen.

2.3 Von der Biologie inspirierte neue Rechnerstrukturen

Die traditionelle Struktur unserer Rechner ist gekennzeichnet durch folgende Merkmale:

- Speicher und Rechnen sind getrennt
- der Rechner kann nur ausführen, was im Programm "vorhergedacht" wurde
- hierarchisch geordnet, sequentielle Arbeitsweise
- Rhythmus durch zentralen Taktgeber vorgegeben
- Kommunikation intern und extern mittels Bits
- Die Prozessoren bestehen aus Gattern, die eine geringe Zahl binärer Inputs verarbeiten.

Im Gegensatz dazu besitzt die Informationsverarbeitung in biologischen Nervensystemen die folgenden charakteristischen Strukturmerkmale:

- Speicher und Rechnen sind kombiniert in den informationsverarbeitenden Grundbausteinen (Neuronen und Synapsen)
- das Nervensystem kann aus Erfahrung lernen
- es ist überwiegend "demokratisch" strukturiert, d.h. es gibt keine zentrale Befehlsausgabe und die Neuronen und Synapsen arbeiten weitgehend autonom
- es gibt keinen zentralen Taktgeber, die zeitliche Abfolge der internen "Rechenschritte" hängt vom konkreten Input ab
- interne Kommunikation und Output mittels spike trains, also zeitlichen Mustern von Pulsen
- Neuronen erhalten durch direkte Verbindungen Signale von 5000 – 10000 anderen Neuronen.

[16] http://www.cs.brown.edu/people/lpk/rl-survey/rl-survey.html/

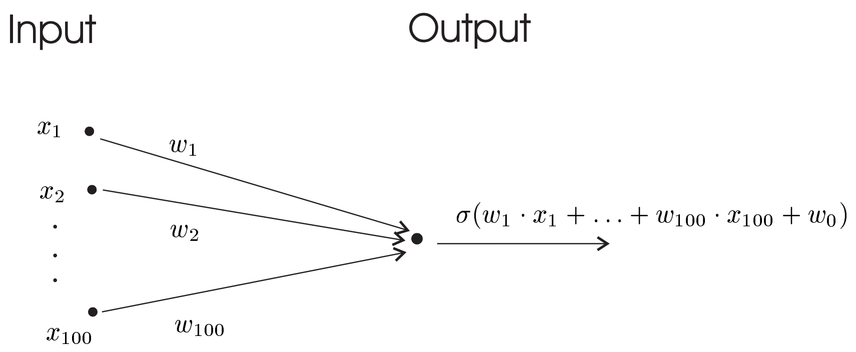

Abbildung 1. Schematische Arbeitsweise eines künstlichen Neurons. Der Input besteht in diesem Fall aus 100 Input Zahlen $x_1, \ldots, x_{100}$ (wobei die Zahl 100 willkürlich gewählt wurde; ein biologisches Neuron erhält bis zu 10 000 Input Zahlen). Der Output besteht aus einer einzigen Zahl zwischen 0 und 1, die in der Zeichnung mit $\sigma(w_1 \cdot x_1 + \ldots + w_{100} \cdot x_{100} + w_0)$ bezeichnet ist. Sie entsteht also dadurch, daß man zunächst die gewichtete Summe $w_1 \cdot x_1 + \ldots + w_{100} \cdot x_{100} + w_0$ berechnet, und auf diese Zahl dann die Quetschungsfunktion σ anwendet.

In der Zwitterwelt zwischen diesen beiden Rechnerstrukturen sind die *künstlichen Neuronalen Netze* angesiedelt. Diese folgen in ihrer Architektur eher den Prinzipien biologischer Systeme, können aber in jedem üblichen digitalen Rechner simuliert, oder direkt in neuartiger elektronischer hardware implementiert werden.[17]Ihre Merkmale sind die folgenden:

- Sie bestehen aus künstlichen Neuronen, die mittels geeigneter Computer-Programme oder sogar mittels dafür entwickelter neuer elektronischer Hardware in einem Computer simuliert werden.
- Memory und Rechnen sind kombiniert in jedem künstlichen Neuron
- das Netzwerk kann aus Erfahrung lernen (mittels einem "Lernalgorithmus")
- es ist überwiegend "demokratisch" strukturiert und hat eine parallele Arbeitsweise
- Rhythmus durch zentralen Taktgeber vorgegeben
- Kommunikation mittels Bits (oder Zahlen).

Die Arbeitsweise eines künstlichen Neurons ist recht einfach. Wir betrachten als Beispiel ein künstliches Neuron mit 100 Input-Variablen $x_1, \ldots, x_{100}$, die Bits oder reelle Zahlen repräsentieren können (siehe Abbildung 1).

[17]Carver Mead, Professor am California Institute of Technology und maßgeblicher Pionier bei der Entwicklung der all unseren gegenwärtigen Rechnern zugrunde liegenden integrierten Schaltkreisen, hat für diese Entwicklungsrichtung den Namen "neuromorphic engineering" geprägt, und in seinem Buch [Mead, 1989] zugleich einen der wichtigsten Beiträge hierzu geliefert.

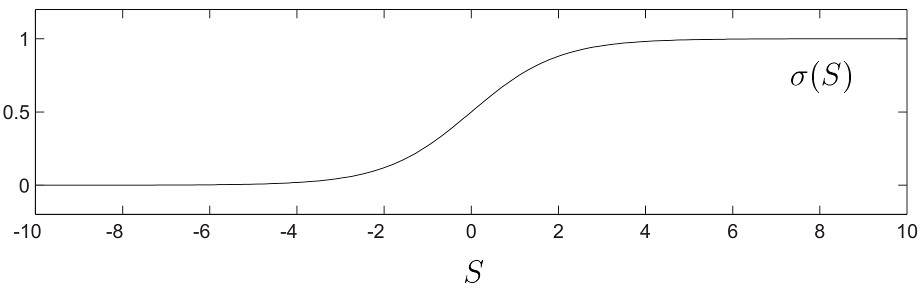

Abbildung 2. Quetschungsfunktion σ (die skizzierte Funktion ist die Funktion $\sigma(S) = 1/(1+e^{-S})$).

Der Output $\sigma(S)$ eines künstlichen Neurons entsteht dadurch, daß es zunächst eine geeignete gewichtete Summe

$$S = w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_{100} \cdot x_{100} + w_0$$

der Inputs $x_1, \ldots, x_{100}$ ausrechnet, und dann eine sogenannte "Quetschungsfunktion" σ auf diese gewichtete Summe S anwendet (zum Beispiel $\sigma(S) = 1/(1+e^{-S})$). Die Gewichte $w_0, \ldots, w_{100}$ sind Zahlen, die das gelernte Wissen des künstlichen Neurons repräsentieren. Zum Beispiel, w_{17} ist eine große positive Zahl, wenn das Neuron die "Erfahrung´´ gemacht hat, daß es einen relativ großen Output-Wert geben sollte, falls sein Input x_{17} groß ist, und w_{51} ist eine negative Zahl, wenn ein hoher Wert von x_{51} zu einem niedrigen Output-Wert des Neurons führen soll (siehe den folgenden Unterabschnitt 2.3.2). Der konstante Term w_0 legt gewissermaßen die "Reizschwelle" des künstlichen Neurons fest, also er legt fest wie hoch die Summe $w_1 \cdot x_1 + \ldots + w_{100} \cdot x_{100}$ werden muß, damit der Output $\sigma(S)$ einen hohen Wert ergibt.
Die nichtlineare Quetschungsfunktion σ transformiert die oft stark positive oder stark negative Zahl S in eine Zahl $\sigma(S)$ zwischen 0 und 1 (Änderung des Maßstabs), siehe Abbildung 2. Diese Änderung des Maßstabs ist *nicht-linear*, weil die Randbereiche mehr zusammengestaucht werden als der Mittelbereich.

2.3.1 Wie verbindet man die "Neuronen" in einem künstlichen Neuronalen Netz?

Ein einzelnes künstliches Neuron kann, selbst bei beliebiger Wahl der Gewichte $w_1, \ldots, w_n$ und der Reizschwelle w_0, nur ein beschränktes Repertoire von Funktionen berechnen. Wenn man diesen Berechnungsvorgang aber iteriert, also den Output von einer Gruppe von Neuronen zum Input eines darauffolgenden weiteren Neurons macht (welches in der Regel andere Gewichte verwendet), so kann das so entstandene Netzwerk von künstlichen Neuronen (siehe Abbildungen 3 und 4) praktisch beliebig komplizierte Abbildungen von Inputs zu Outputs berechnen, oder vielmehr zu berechnen "lernen", wie wir im folgenden Unterabschnitt sehen.

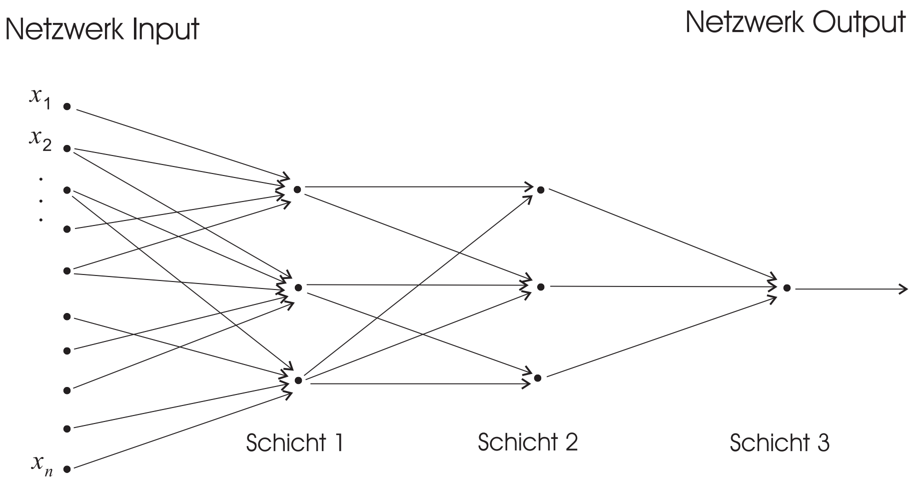

Abbildung 3. Verbindungsstruktur eines typischen in einer Richtung ausgerichteten ("feed-forward") Neuronalen Netzes. Die Richtung der Berechnung in diesem Neuronalen Netz geht von links nach rechts. Jeder Punkt auf den Schichten 1,2,3, zusammen mit den in diesen Punkt gerichteten Pfeilen, symbolisiert ein künstliches Neuron wie in Abb. 1. Die Inputs der Neuronen auf Schicht 1 bestehen aus Netzwerk Inputs $x_1, \ldots, x_n$. Im 1. Zeitschritt berechnen alle Neuronen auf Schicht 1 gleichzeitig ("parallel") ihre jeweilige Output Zahl. Die drei von den Neuronen auf Schicht 1 berechneten Output Zahlen werden weiter verarbeitet von Neuronen auf Schicht 2, d.h. sie liefern die Inputs für Neuronen auf Schicht 2. Die Neuronen auf Schicht 2 errechnen ihre Output Zahlen während dem 2. Zeitschritt, und aus diesen berechnet dann das Neuron auf Schicht 3 seinen Output, der gleichzeitig den Output des gesamten Neuronen Netzes (bestehend aus 7 Neuronen) bildet.

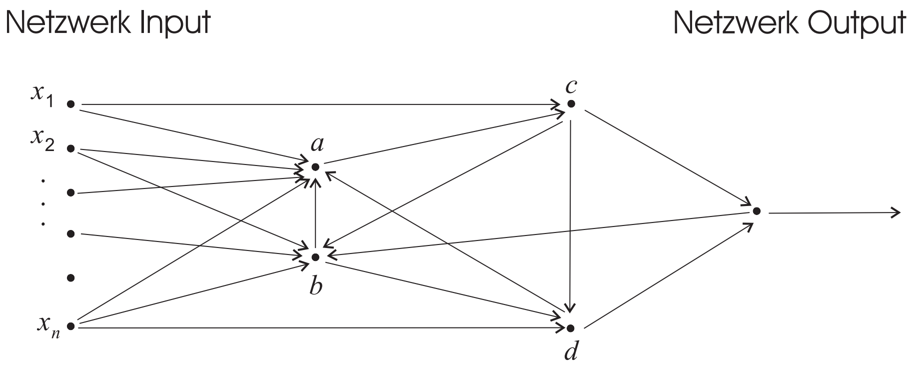

Abbildung 4. Beispiel für ein Neuronales Netz mit einer komplexeren Verbindungsstruktur. Der Netzwerk Input ist wieder links abgebildet, und der Netzwerk Output ist wieder rechts. Intern ist der Informationsfluß aber komplizierter. Das Neuron a erhält als Inputs nicht nur Netzwerk Inputs, sondern auch die Outputs der Neuronen b und d. Die Neuronen b und d bekommen als Input unter anderem den Output von Neuron c, das wiederum den Output von Neuron a als Input bekommt. Wegen der vielfältigen "Rückkopplung" im Netz ist es nicht mehr so einfach zu beschreiben, wie die einzelnen Rechenschritte im Netz koordiniert werden. Komplizierte Netzwerk-Strukturen mit "Rückkopplungen" sind typisch für biologische Netzwerke von Neuronen, werden aber für eine große Anzahl von Anwendungen auch künstlich im Rechner simuliert.

2.3.2 Woher kommen die "Gewichte" in einem künstlichen Neuronalen Netz?

Die Gewichte und Reizschwellen w_i der künstlichen Neuronen in einem Neuronalen Netz sind Zahlen, die das "Programm" eines Neuronalen Netzes enthalten, also seine Arbeitsweise festlegen. Anstatt die Werte der Gewichte und Reizschwellen fest einzuprogrammieren, überläßt man es dem Neuronalen Netz, deren Werte selbst so zu wählen, daß es das gewünschte Input/Output-Verhalten zeigt. Dazu wird es mit Trainingsbeispielen für "gute" Kombinationen von Inputs und Outputs gefüttert. Mittels einem einprogrammierten Lernalgorithmus kann es diese Trainingsbeispiele benutzen um seine Gewichte und Reizschwellen, und damit seine Arbeitsweise, in geeigneter Weise zu verändern.
Man verwendet unter anderem die folgenden Design Ideen bei der Konstruktion von Lernalgorithmen für künstliche neuronale Netze:

- Ermittle für jedes Gewicht in welcher Richtung man es verändern soll, damit das Verhalten des Neuronalen Netzes "in die richtige Richtung" verändert wird
- Geduld: Viele richtige kleine Schritte bei Änderungen von Gewichten und Reizschwellen führen auch zum Ziel
- Sei gelegentlich "kreativ", z.B. probier einfach aus, was passiert, wenn man einem Gewicht einen ganz anderen Wert gibt ($\rightarrow$ Zufallselemente).

2.3.3 Was leistet ein künstliches Neuronales Netz?

Mit genügend vielen Trainingsbeispielen kann ein Neuronales Netz zum Beispiel lernen

- handgeschriebene Zeichen zu lesen
- dem Arzt bei der Diagnose von Krankheiten zu helfen (z.B. Früherkennung von Brustkrebs)
- einen Motor so zu steuern, daß der Ausstoß von Schadstoffen minimiert wird
- einen Roboter so zu steuern, daß er Kollisionen mit Hindernissen vermeidet.

Abgesehen von ihrer Lernfähigkeit haben Neuronale Netze den Vorteil, daß alle praktisch wichtigen Funktionen mittels in einer Richtung ausgerichteten Netzen bestehend aus nur 2 Schichten (und daher in nur 2 parallelen Rechenschritten) mit nicht zu vielen künstlichen Neuronen berechnet werden können. Daher ergeben Neuronale Netze gleichzeitig ein sehr nützliches mathematisches Modell für Rechnen in "Echtzeit".[18]

[18] Eine schöne Einführung in künstliche Neuronale Netze ist das Tutorial von Gerstner "Supervised Learning for Neural Networks: A Tutorial with JAVA exercises": http://diwww.epfl.ch/lami/team/gerstner/wg_pub.html
Einige links zu weiterer Literatur, Forschungsberichten und Simulatoren für Neuronale Netzwerke findet man unter http://www.cis.tu-graz.ac.at/igi/maass/♯Some Links, http://diwww.epfl.ch/w3mantra/mantra_links.html, http://diwww.epfl.ch/w3mantra/mantra_journals.html

2.4 Wie unterscheidet sich ein künstliches Neuronales Netz von seinem Vorbild in der Natur?

Wenn man die Arbeitsweise und Leistung der gegenwärtigen Generation von künstlichen Neuronalen Netzen mit der von biologischen Nervensystemen vergleicht, stellt sich heraus, daß sie weiter voneinander entfernt sind, als man früher gedacht hatte. Ein wesentlicher Unterschied zwischen biologischen und künstlichen Neuronen liegt in der Struktur ihres Outputs. Der Output eines biologischen Neurons besteht aus elektrischen Pulsen ("spikes"), die in unregelmäßigen Abständen, ca 1–100 mal pro Sekunde, ausgesendet werden. Wenn man die Zeiten protokolliert,

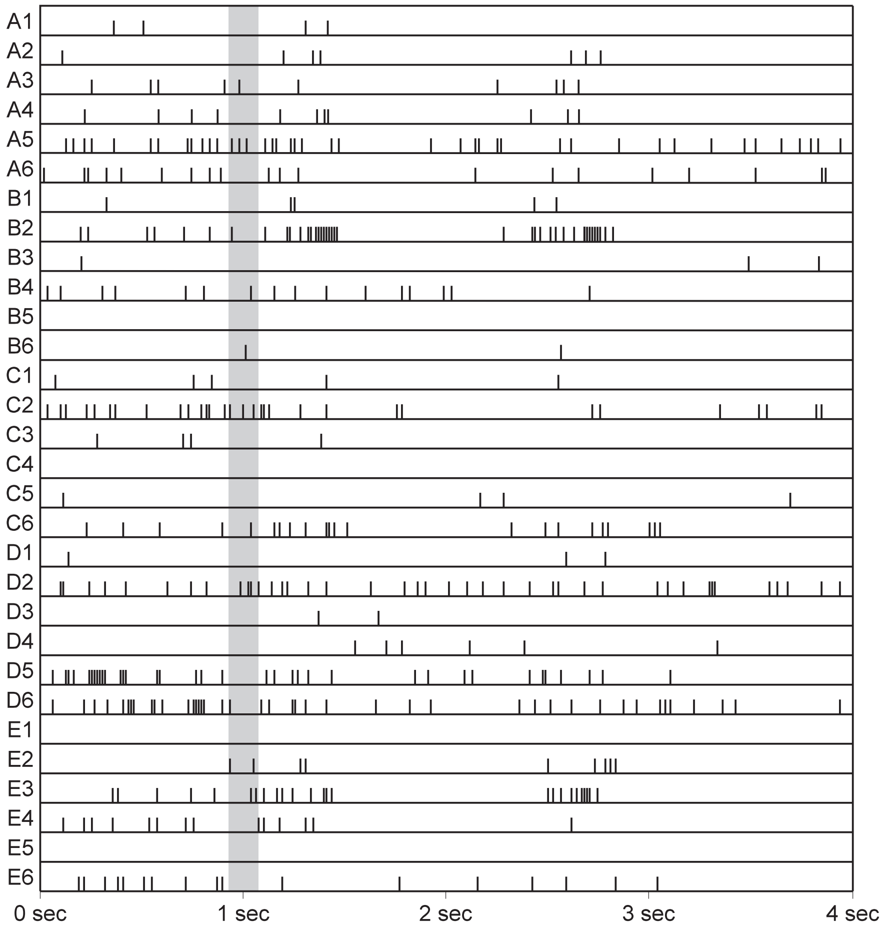

Abbildung 5. Aufnahme der Feuerzeiten von 30 Neuronen in der visuellen Kortex des Affen über 4 Sekunden [Krüger and Aiple, 1988]. Die 30 Neuronen wurden von den Experimentatoren mit A1 bis E6 bezeichnet (siehe linke Spalte der Abbildung). Das Feuerverhalten von jedem der 30 Neuronen ist jeweils in einer Zeile protokolliert. Die waagrechte Achse ist die Zeitachse. Jede Feuerzeit ist durch einen senkrechten Strich angezeigt. Man bezeichnet die jeweils in einer Zeile festgehaltene Folge von Feuerzeichen eines Neurons als "spike train". Zum Vergleich mit der Zeitdauer einer typischen Berechnung in einem Nervensystem haben wir ein Zeitintervall von 150 Millisekunden grau markiert. Innerhalb dieser Zeitdauer können biologische Nervensysteme (bestehend aus 10 und mehr Schichten) komplexe Aufgaben der Musterkennung ausführen. Abdruck mit freundlicher Genehmigung von The Amercian Physiological Society.

wann ein biologisches Neuron einen spike aussendet, so schaut ein solcher "spike train" so aus wie eine der Zeilen von Abbildung 5. In Abbildung 5 wurden die spike trains von 30 (ziemlich willkürlich ausgewählten) Neuronen für einen Zeitraum von 4 Sekunden aufgezeichnet. Wenn wir zum Beispiel alle Informationen protokollieren würden, die unser Gehirn innerhalb von 4 Sekunden von unseren Augen erhält, so würde eine ähnliche Abbildung mit 1.000.000 Zeilen (anstatt 30) entstehen, weil alle visuellen Eindrücke in der Retina in die spike trains von 1.000.000 Neuronen kodiert und in dieser Form über den optischen Nerven an das Gehirn weitergeleitet werden.

Man hat früher gemeint, daß das einzige relevante Signal im Output eines biologischen Neurons die *Häufigkeit* seines Feuerns ist, und daß diese Häufigkeit dem Output $\sigma(S)$ eines künstlichen Neurons entspricht. Man sieht aber sofort aus Abbildung 5, daß sich die momentane Häufigkeit des Feuerns eines biologischen Neurons ständig ändert, und daß die zeitlichen Abstände zwischen dem Feuern eines Neurons viel zu unregelmäßig sind um aus 2 oder 3 spikes eine gute Abschätzung von dessen gegenwärtiger Häufigkeit des Feuerns zu ermöglichen. Neuere experimentelle Untersuchungen (siehe zum Beispiel [Rieke et al., 1997, Koch, 1999, Recce, 1999]) zeigen vielmehr, daß das gesamte raum-zeitliche Muster des Feuerns von biologischen Neuronen für deren Informationsverarbeitung relevant ist. Man kann den Output eines Systems von biologischen Neuronen also eher mit einem von einem großen Orchester gespielten Musikstück vergleichen, zu dessen Wiedererkennung es nicht ausreicht zu wissen *wie oft* jedes Instrument gewisse Töne spielt. Charakteristisch für ein Musikstück ist vielmehr, wie jeder Ton in eine Melodie oder in einen Akkord eingebettet ist. Man nimmt an, daß in ähnlicher Weise viele Gruppen von Neuronen in biologischen Systemen die von ihnen ausgesendeten Informationen durch das zeitliche Muster kodieren, in dem jedes Neuron in der Gruppe *relativ zu den anderen* feuert. Daher ist die Kommunikationsweise innerhalb unseres Gehirns einem Musikstück sehr viel ähnlicher als die von der gegenwärtigen Generation von Computern bevorzugte Kommunikationsweise.
Die Untersuchung der theoretischen und praktischen Möglichkeiten mittels raumzeitlicher Muster von Pulsen zu rechnen und zu kommunizieren, hat in den letzten Jahren eine neue Generation künstlicher Neuronaler Netzwerke enstehen lassen: *pulsbasierte* künstliche Neuronale Netze.[19] In diesen wird die künstliche Synchronisation der traditionellen künstlichen Neuronalen Netzen ersetzt durch die der Biologie abgeschaute Methode, Informationen in raum-zeitlichen Mustern (analog wie bei einem Handzeichen oder einer Melodie) zu kodieren. Dies eröffnet für den Informatiker eine faszinierende neue Welt, nämlich die Möglichkeit *Zeit* als eine bisher in unseren Rechnern brachliegende Dimension zu erschließen, und direkt *mit raumzeitlichen Mustern zu rechnen* [Maass, 1999, Maass and Sontag, 1999].[20]
Erfreulicherweise kann man diese Strategie auch relativ leicht in neu entwickelter elektronischer Hardware anwenden [Murray, 1999].[21] Diese Kodierungsstrate-

[19][Maass and Bishop, 1999] enthält Übersichtsartikel zum gegenwärtigen Stand der Forschung über pulsbasierte Neuronale Netze.

[20]Einige Arbeiten zu diesem Thema sind online erhältlich von http://www.cis.tu-graz.ac.at/igi/maass/Welcome.html.

[21]Im Prinzip kann man sich auch damit behelfen, daß man so einen neuartigen Rechner auf einem

gie implementiert eine faszinierende Idee von [Mead, 1989]: "to let time represent itself", d.h. anstatt die zeitliche Struktur der Inputs (zum Beispiel visuelle Informationen über bewegte Objekte) durch angehängte künstliche "time-stamps" – vergleichbar den Eingangsstempeln in einem Büro – zu kodieren, bleibt die zeitliche Struktur des Inputs während des "Rechnens" in der raum-zeitlichen Struktur der spike trains repräsentiert. In einer Analogie kann man diese Vorgangsweise vergleichen mit der Arbeitsweise in der Küche eines (noch nicht vollständig digitalisierten) Restaurants. Dort werden die vom Ober aufgenommenen Bestellzettel für die Köche gut sichtbar an einem Rad (oder gelegentlich an einer Schiene) befestigt. In diesem Fall kodiert die Position jedes einzelnen Bestellzettels relativ zu den anderen die Zeit seines Eingangs. Auch ohne Eingangsstempel ("time stamps") auf den Bestellzetteln können bei dieser raum-zeitlichen Organisationsstruktur die Bestellungen in der richtigen Reihenfolge ausgeführt werden. Der Ansatz, daß man traditionelle Rechenschritte ersetzt durch geeignete Manipulationen von raum-zeitlichen Strukturen, eröffnet neue Möglichkeiten um zum einen Energie-effiziente parallele Rechner zur Verarbeitung komplexer Information in Echtzeit zu entwerfen, und zum anderen möglicherweise ein wenig besser zu verstehen wie komplexe raum-zeitliche Informationen im menschlichen Gehirn kodiert und verarbeitet werden.[22]

3 Konklusion

Ich habe die Frage nach der Informationsverarbeitung im menschlichen Gehirn bisher ausgeklammert. Dies geschah zum einen deshalb, weil ich kein Experte für dieses Gebiet bin, zum anderen, weil Gespräche mit Neurophysiologen und Neurobiologen deutlich machen, daß über die Informationsverarbeitung im menschlichen Gehirn sehr viel weniger gesichertes Wissen vorliegt, als in der typischen Populär-Literatur suggeriert wird. In manchen populärwissenschaftlichen Artikeln liest man, daß es schon jetzt möglich wäre, mittels geeigneter Apparate "Gedanken zu lesen" im menschlichen Gehirn, während andere Artikel das bestreiten. Darüberhinaus vermitteln zahlreiche Veröffentlichungen in wissenschaftlichen Fachzeitschriften den Eindruck, daß die wesentlichen Rätsel bezüglich der Informationsverarbeitung in biologischen Nervensystemen bereits gelöst seien. Leider gibt es zu fast allen solchen Veröffentlichungen andere, genauso seriöse Veröffentlichungen, die behaupten, dieselben Rätsel mit vollkommen anderen Antworten gelöst zu haben, oder daß diese Rätsel noch ungelöst wären.
Dieser verwirrende Zustand wird verständlich wenn man sich die experimentellen Ergebnisse genauer anschaut, die diesen Artikeln zugrunde liegen. Wir erleben

herkömmlichen digitalen Rechner simuliert [Jahnke et al., 1999]. Praktisch scheitert dies oft daran, daß die erforderliche Rechenzeit zu groß ist. Daher schlägt man stattdessen an der ETH Zürich den von Carver Mead vorgezeichneten Weg ein und baut eine analoge "silicon cortex" [Deiss et al., 1999].

[22]Für weitere Informationen über pulsbasierte Neuronale Netze verweise ich auf die allgemeinverständliche Einführung von Natschläger in deutscher Sprache: http://www.cis.tu-graz.ac.at/igi/tnatschl/3gen/3genlang.html
sowie auf die detaillierten Übersichtsartikel "Spiking neurons" von Gerstner
http://diwww.epfl.ch/lami/team/gerstner/wg_pub.html
sowie auf meinen Übersichtsartikel "Paradigms for Computing with Spiking Neurons" von
http://www.cis.tu-graz.ac.at/igi/maass/♯Publications.

gegenwärtig gewaltige Fortschritte bei *nichtinvasiven* Methoden zur Messung von Gehirnaktivitäten, wie zum Beispiel EEG, PET, MRI (für allgemeinverständliche Beschreibungen dieser Techniken siehe [Schmidt, 1993, Purves et al., 1997]). Mit diesen Methoden erhält man raum-zeitliche Muster von Gehirnaktivitäten, wobei aber die räumliche Auflösung so gering ist, daß man nur Informationen über das durchschnittliche Aktivitätsniveau von neuronalen Schaltkreisen bestehend aus Milliarden von Neuronen erhält, aber nicht erfährt wie dort im Einzelnen Informationen kodiert und verarbeitet werden. Man kann aber mittels solcher Methoden zum Beispiel feststellen, ob die Gehirnregion, die für die Steuerung der rechten Hand zuständig ist, überdurchschnittlich aktiviert ist. Da die für die Steuerung der rechten Hand zuständige Gehirnregion schon dadurch aktiviert werden kann, daß die betreffenden Person nur intensiv daran *denkt* die rechte Hand zu bewegen, kann man diesen *Gedanken* in der Regel im EEG der betreffenden Person ablesen. Es ist aber nicht vorstellbar, daß man mit solchen Apparaten abstraktere Gedanken eines Menschen lesen kann, also zum Beispiel an welche Zahl eine Person gerade denkt; oder welche Partei sie am kommenden Wahlsonntag wählen möchte.
Mittels *invasiver* Methoden erhält man bei Tieren zusätzlich Informationen über das genaue Feuerverhalten einer kleineren Anzahl von Neuronen – siehe zum Beispiel Abbildung 5. Aber hier hat man das Problem, daß unbekannt bleibt, welchen *Input* diese Neuronen gerade von ihren bis zu 10 000 "Vorgänger"-Neuronen bekommen, und welche intrazellulären biochemischen Substanzen ihr gegenwärtiges Verhalten beeinflußen. Daher erhält man bei diesen Experimenten nur in recht indirekter Weise Informationen über die von diesen Neuronen ausgeführten Berechnungen. In wirbellosen Tieren gelingt es gelegentlich, *einzelne* Neuronen zu identifizieren, die eine Schlüsselrolle bei der internen Kodierung von sensorischem Input (also zum Beispiel von dem, was das Auge sieht) spielen. In einzelnen Fällen, wie zum Beispiel bei den beiden H1-Neuronen in der Fliege, konnte sogar der "Code" geknackt werden, mit dem diese Information im Feuerverhalten dieser Neuronen kodiert ist [Rieke et al., 1997, Recce, 1999] .
Leider ist es bisher nicht einmal bei relativ einfachen Tieren gelungen, den "Neuronalen Kode" zu identifizieren, mit dem *Zwischenergebnisse* bei der weiteren Verbreitung der sensorischen Informationen kodiert sind, sodaß selbst hier der Zugang zum Verständnis der eigentlichen Informationsverarbeitung in den darauffolgenden neuronalen Schaltkreisen versperrt bleibt. Bei höheren Tieren ist die interne neuronale Kodierung noch sehr viel komplexer, weil Informationen in verteilter Weise kodiert sind, also aufgespalten über sehr große Gruppen von Neuronen.
Ein Gedankenexperiment hilft uns vielleicht, die Schwierigkeit zu verstehen, aus experimentellen Daten der vorher geschilderten Art auf die Organisation der Informationsverarbeitung im menschlichen Gehirn zu schließen. Nehmen wir einmal an, ein künstlicher Rechner der gegenwärtigen Generation wäre aufgrund eines time-warps versehentlich im Jahr 1920 "vom Himmel gefallen", und man hätte in einem Forschungslabor mittels aller damals bekannten Meßmethoden zu ermitteln versucht, wie die Informationsverarbeitung in dieser Maschine funktioniert. Mittels Messungen der Temperatur an verschiedenen Stellen dieser Maschine (welche Informationen liefern über den lokalen Energieverbrauch, analog zu PET und MRI beim Gehirn) und Messungen elektrischer Felder (analog zum EEG) hätte man vielfältige Aktivitätsmuster und deren Abhängigkeit von Eingaben über die Tasta-

tur studieren können. Zusätzlich hätte man vielleicht auch das zeitliche Muster des Schaltens einzelner Transistoren (entsprechend dem Feuerverhalten einzelner Neuronen im Gehirn) ermitteln können. Trotzdem glaube ich nicht, daß man mittels solcher "bottom-up Methoden" die Struktur des Betriebssystems dieses Computers hätte ermitteln können. Vielmehr hätte man diese bottom-up Methoden ergänzen müssen durch top-down Modelle, bei denen man fortschreitend feinere und adäquatere Hypothesen über die Organisationsstruktur des Betriebssystems entwickelt, und in künstlichen Maschinen zur Informationsverarbeitung erprobt. In anderen Worten: man hätte die Funktionsweise des im Jahr 1920 vom Himmel gefallenen Rechners nur dann ermitteln können, wenn man begleitend zu den experimentellen Untersuchungen wesentliche Teile der Rechner-Theorie vorausgedacht und an entsprechend konstruierten Prototypen *ausprobiert* hätte.

Im Vergleich zu diesem Gedankenexperiment ist das Problem der Entschlüsselung der Arbeitsweise des menschlichen Gehirns ungleich größer. Anstatt durch begrifflich strukturierte Design-Prinzipien von menschlichen Ingenieuren ist das menschliche Gehirn durch die Evolution, also im Laufe einer langen Kette von großteils zufallsgesteuerten trial-and-error Experimenten geformt worden. Daher ist es hier noch schwieriger als im vorhergehenden Gedankenexperiment, in direkter Weise experimentell zu ermitteln, wie Berechnungen strukturiert sind. Deshalb muß man auch im Bereich der Neurowissenschaften die bottom-up Methode durch top-down Ansätze ergänzen, bei denen man versucht die vielen bruchstückhaften aus Experimenten gewonnenen Erkenntnisse, Indizien und Vermutungen mittels *theoretischer Modelle und Computersimulationen* zu globalen Hypothesen über die Struktur neuronaler Berechnungen in konkreten Lebewesen zu kombinieren. Jede solche globale Hypothese kann gleichzeitig aufgefaßt werden als grober Bauplan für die Organisation eines (möglicherweise stark spezialisierten) künstlichen Rechners, und die Tragfähigkeit solcher Hypothesen kann eigentlich nur dadurch ermittelt werden, daß man solch einen neuartigen künstlichen Rechner baut.

Man sieht also, daß die Entschlüsselung der Informationsverarbeitung im menschlichen Gehirn und die Entwicklung intelligenter Rechner Hand in Hand gehen. In der Tat sind alle im 2. Abschnitt diskutierten innovativen Ideen zur Entwicklung "intelligenter" Rechner aus dem Bemühen, Informationsverarbeitung in konkreten biologischen Nervensystemen zu verstehen (siehe [Arbib, 1995]), entstanden. Aus dem vermeintlichen Gegensatz zwischen unseren Erkenntnissen über das menschliche Gehirn und dem Entwurf von leistungsfähigen künstlichen Rechnern entsteht also bei genauerem Hinschauen eine Symbiose von zwei sehr verschiedenartigen Wissensbereichen, wobei Fortschritte im Verständnis des ersten Bereichs untrennbar verbunden sind mit innovativen technischen Ideen im zweiten Bereich. In anderen Worten: *Je mehr wir von der Organisation der Informationsverarbeitung im menschlichen Gehirn verstehen werden, umso "gehirnartiger" werden gleichzeitig die besten künstlichen Rechner werden, und umso weniger befremdlich wird die Vorstellung sein, daß das menschliche Gehirn "nur" ein Rechner sei.*

Nachsatz: Ich möchte Peter Auer, Margot Goettsberger, Gerold Muhr und Thomas Natschläger für hilfreiche Kommentare zum ersten Entwurf dieses Aufsatzes danken.

Literatur

[Arbib, 1995] Arbib, M. A., editor (1995). *The Handbook of Brain Theory and Neural Networks*. MIT-Press (Cambridge, MA).

[DePauli und Weibel, 1997] DePauli-Schimanovich, W., Weibel, P. (1997). *Kurt Gödel: ein Mathematischer Mythos*. Hölder-Pichler-Tempsky (Wien).

[Deiss et al., 1999] Deiss, S. R., Douglas, R. J., and Whatley, A. M. (1999). A pulse-coded communications infrastructure for neuromorphic systems. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Jahnke et al., 1999] (1999). Digital Simulation of Spiking Neural Networks. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press (Cambridge, MA).

[Koch, 1999] Koch, C. (1999). *Biophysics of Computation: Information Processing in Single Neurons*. Oxford University Press (Oxford).

[Krüger and Aiple, 1988] Krüger, J., and Aiple, F. (1988). Multielectrode investigation of monkey stritate cortex: Spike train correlations in the infragranular layers. *Neurophysiology*, 60:798–828.

[Maass, 1999] Maass, W. (1999). Computing with spiking neurons.In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press (Cambridge, MA).

[Maass and Bishop, 1999] Maass, W., and Bishop, C., editors (1999). *Pulsed Neural Networks*. MIT-Press (Cambridge, MA).

[Maass and Sontag, 1999] Maass, W., and Sontag, E. D. (1999). *Neural systems as nonlinear filters*, submitted for publication.

[Mead, 1989] Mead, C. (1989). *Analog VLSI and Neural Systems*. Addison-Wesley (Reading).

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*, McGraw-Hill (New York).

[Murray, 1999] Murray, A. F. (1999). Pulse-based computation in VLSI neural networks. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press (Cambridge, MA).

[Picard, 1997] Picard, R. W. (1997). *Affective Computing*, MIT Press (Cambridge, MA).

[Purves et al., 1997] Purves, D., Augustine, G. J., Fitzpatrick, D., Katz, L. C., La Mantia, A.-S., McNamara, J. O., eds. (1997). *Neuroscience*, Sinauer Associates (Sunderland, MA).

[Recce, 1999] Recce, M. (1999). Encoding Information in Neuronal Activity. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press (Cambridge, MA).

[Rieke et al., 1997] Rieke, F., Warland, D., Bialek, W., and de Ruyter van Steveninck, R. (1997). *SPIKES: Exploring the Neural Code*. MIT-Press (Cambridge, MA).

[Schmidt, 1993] Schmidt, R. F. (1993). *Neuro- und Sinnesphysiologie*, Springer (Berlin).

[Sipser, 1997] Sipser, M. (1997). *Introduction to the Theory of Computation*, PWS Publishing Company (Boston, MA).

[Sutton and Barto, 1998] Sutton, R. S., Barto, A. G. (1998). *Reinforcement Learning*, MIT Press (Cambridge, MA).

[Turing, 1950] Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59, no. 236.

[Weiss and Kulikowski, 1991] Weiss, S. M., Kulikowski, C. A. (1991). *Computer Systems that Learn*, Morgan Kaufmann (San Mateo, CA)